

AI PLANNING ASSISTANT FOR SCHEDULING DAILY
ACTIVITIES

Priyanka Ahuja
2018

Columbus State University

The D. Abbott Turner College of Business

The Graduate Program in Applied Computer Science

AI Planning Assistant for Scheduling Daily Activities

A Thesis in

Applied Computer Science

by

Priyanka Ahuja

Submitted in Partial Fulfillment

Of the Requirements

For the Degree of

Master of Science

May 2018

©2018 by Priyanka Ahuja

TABLE OF CONTENTS

I have submitted this thesis in partial fulfillment of the requirements for the degree of Master of Science

ACKNOWLEDGEMENT

Chapter 1: Introduction

| | |
|-----------------------|---|
| 1.1 Problem Statement | 1 |
|-----------------------|---|

| | |
|-----------------------|---|
| 1.2 Literature Review | 2 |
|-----------------------|---|

Date

Priyanka Ahuja

| | |
|----------------------|---|
| 1.3 Our Contribution | 6 |
|----------------------|---|

| | |
|-------------------------|---|
| 1.4 Thesis Organisation | 6 |
|-------------------------|---|

We approve thesis of Priyanka Ahuja as presented here;

Chapter 2: Background

| | |
|------------------|---|
| 2.1 Introduction | 7 |
|------------------|---|

| | |
|--|---|
| 2.2 Recent Trends in the development of Schedule Tools | 8 |
|--|---|

| | |
|--|----|
| 2.3 Natural Language Processing and Natural Language | 10 |
|--|----|

Date

Rania Hodhod, Ph.D.

| | |
|--------------------------------|----|
| 2.4 Human-Computer Interaction | 11 |
|--------------------------------|----|

Assistant Professor of Computer Science,
Thesis Advisor

| | |
|-------------------------|----|
| 2.4.1 Types of Chatbots | 13 |
|-------------------------|----|

| | |
|-------------------------------------|----|
| 2.4.2 Chatbot Development Platforms | 15 |
|-------------------------------------|----|

| | |
|------------------|----|
| 2.5 Web Services | 17 |
|------------------|----|

Date

Shamim Khan, Ph.D.

Chapter 3: Architecture of the System

Professor of Computer Science

| | |
|------------------|----|
| 3.1 Introduction | 21 |
|------------------|----|

| | |
|-------------------------|----|
| 3.2 System Architecture | 21 |
|-------------------------|----|

| | |
|--------------|----|
| 3.3 Training | 24 |
|--------------|----|

| | |
|---------------|----|
| 3.3.1 Actions | 25 |
|---------------|----|

Date

Alfredo Perez, Ph.D.

| | |
|----------------|----|
| 3.3.2 Response | 26 |
|----------------|----|

Assistant Professor of Computer Science

| | |
|-----------------|----|
| 3.3.3. Contexts | 26 |
|-----------------|----|

| | |
|-------------------|----|
| 3.4 External Aops | 27 |
|-------------------|----|

| | |
|-------------------------------|----|
| 3.4.1 HERE Predictive Traffic | 27 |
|-------------------------------|----|

| | |
|-------------------------|----|
| 3.4.2 Yahoo weather App | 27 |
|-------------------------|----|

Date

Wayne Summers, Ph.D.

Chapter 4: System Evaluation and Results

Distinguished Chairperson
Professor of Computer Science

| | |
|------------------|----|
| 4.1 Introduction | 28 |
|------------------|----|

| | |
|-------------------------------------|----|
| 4.2 Evaluation Metrics for Schedule | 28 |
|-------------------------------------|----|

| | |
|---------------------------------|----|
| 4.2.1 Failure Cases in Schedule | 29 |
|---------------------------------|----|

| | |
|------------------|----|
| 4.2.2 System Run | 34 |
|------------------|----|

| | |
|-------------|----|
| 4.3 Results | 36 |
|-------------|----|

TABLE OF CONTENTS

| | |
|---|-------------|
| ABSTRACT | v |
| LIST OF FIGURES | vii |
| ACKNOWLEDGEMENT | viii |
| Chapter 1: Introduction | |
| 1.1 Problem Statement | 1 |
| 1.2 Literature Review | 2 |
| 1.3 Our Contribution | 6 |
| 1.4 Thesis Organization | 6 |
| Chapter 2: Background | |
| 2.1 Introduction..... | 7 |
| 2.2 Recent Trends in the development of Scheduling Tools | 8 |
| 2.3 Natural Language Processing and Natural Language Understanding..... | 10 |
| 2.4 Human-Computer Interaction..... | 12 |
| 2.4.1 Types of Chatbots..... | 13 |
| 2.4.2 Chatbot Development Platforms | 15 |
| 2.5 Web Services | 17 |
| Chapter 3: Architecture of the System | |
| 3.1 Introduction..... | 21 |
| 3.2 System Architecture | 21 |
| 3.3 Training..... | 24 |
| 3.3.1 Actions | 25 |
| 3.3.2 Response | 26 |
| 3.3.3. Contexts..... | 26 |
| 3.4 External Apps..... | 27 |
| 3.4.1 HERE Predictive Traffic | 27 |
| 3.4.2 Yahoo weather App | 27 |
| Chapter 4: System Evaluation and Results | |
| 4.1 Introduction..... | 28 |
| 4.2 Evaluation Metrics for Schedulio | 28 |
| 4.2.1 Failure Cases in Schedulio | 29 |
| 4.2.2 System Run | 34 |
| 4.3 Results | 36 |

Chapter 5: Appendices

| | |
|-------------------------|-----------|
| 5.1 Appendix A | 44 |
| References | 50 |

humans do. Siri, Cortana, and Alexa are examples of intelligent agents that can help us with almost all the basic tasks. These agents are smart enough to do the basic tasks, but not as much when it comes to complex tasks, such as analyzing traffic data, reviewing scheduling conflicts, rescheduling meetings while resolving conflicts, and offering suggestions based upon data analyses (e.g. traffic patterns, weather, etc.). The actual potential of dialogue-based task agent potential remains untapped. The reason is the fact agents lack the ability to fully understand human language. Natural Language processing (NLP) makes it possible for those agents to understand humans.

The current research project explores the development of a conversational agent, Schedullo, that can be connected to the user's calendar and assist with the scheduling, deletion, and modification of events using natural language conversation. Additionally, current research explores the implementation of a recommendation agent that provides suggestions based upon time and slot date recommendation, location-specific weather data, and traffic patterns.

Keywords: Natural Language Processing, Natural Language Understanding, Scheduling Agent.

ABSTRACT

Artificial conversational agents are software agents that can interact with humans in the way humans do. Siri Cortana, and Alexa are examples of intelligent agents that can help us with almost all the basic tasks. These agents are smart enough to do the basic tasks, but not as much when it comes to complex tasks, such as analyzing traffic data, reviewing scheduling conflicts, rescheduling meetings while resolving conflicts, and offering suggestions based upon data analyses (e.g. traffic patterns, weather ...etc.) The actual potential of dialogue-based task agent potential remains untapped. The reason is the fact agents lack the ability to fully understand human language. Natural Language processing (NLP) makes it possible for those agents to understand humans.

The current research project explores the development of a conversational agent, Schedulio, that can be connected to the user's calendar and assist with the scheduling, deletion, and modification of events using natural language conversation. Additionally, current research explores the implementation of a recommendation agent that provides suggestions based upon time and slot date recommendation, location-specific weather data, and traffic patterns.

Keywords: Natural Language Processing, Natural Language Understanding, Scheduling Agent.

LIST OF TABLES

| | |
|--|----|
| Table 4.2.1 Slot-Filling | 30 |
| Table 4.2.2 Evaluation Metrics for Schedulio | 38 |

LIST OF FIGURES

| | |
|---|----|
| Fig. 1 Important stages of human-computer interaction | 12 |
| Fig. 2 Slack platform | 19 |
| Fig. 3 Schedulio Architecture | 22 |
| Fig. 4 Training phase of Dialog Flow Agent | 24 |
| Fig. 5 Actions and Parameters | 26 |
| Fig. 6 Small talk with Dialog flow agent | 33 |
| Fig. 7 Scheduling a meeting using Schedulio | 33 |
| Fig. 8 Confirmation Notification from Schedulio | 34 |
| Fig. 9 Google Calendar Sign-in | 35 |
| Fig. 10 Google Calendar sign-in Confirmation | 35 |
| Fig. 11 Schedule added to Google Calendar | 36 |

LIST OF TABLES

| | |
|--|----|
| Table 4.2.1 Slot-Filing | 30 |
| Table 4.3.1 Evaluation Metrics for Schedulio | 38 |

I would like to thank my advisor Dr. Ranja Hedhod and committee members Dr. Alfredo Perez,

and Dr. Shamim Khan. I would like to thank my husband and family for their endless support.

Chapter 1

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Rania Hodhod and committee members Dr. Alfredo Perez, and Dr. Shamim Khan. I would like to thank my husband and family for their endless support.

Chapter 1

Introduction

1.1 Problem Statement:

Time Management is extremely important in everybody's life. Scheduling is one of the important parts of time management. Unlike planning challenges, scheduling challenges are temporally dependent. Automatic task management systems allow scheduling events in the user's calendar and can suggest alternative schedules if needed. Google Calendar and many other calendars in the market help us to organize tasks and plan our meetings. But all these calendars are not "intelligent" in the sense that they cannot handle conflicts nor provide recommendations. Moreover, they cannot understand or communicate with their users through natural language. This raises the need for smarter calendars that can allow the users to interact with the calendar using natural language.

The aim of this research is to build a task based chatbot for scheduling meetings with recommendations based on weather and traffic information. The purpose of this project is to develop an agent that will assist users in the management of daily tasks and meetings. This functionality will be facilitated through the addition of appointments and tasks to a Google Calendar interface via natural language processing text-based input method. The agent is deployed on Slack, a popular cloud-based team collaboration tool. The conversational agent will be able to analyze the historical traffic information and inform the user of the estimated traffic density in that area. The success of this agent will be determined by the level of linguistic

complexity that the agent is able to successfully handle, that is percentage of utterances that are accurately created as events into the user's Google Calendar account.

This agent was built to help users with organizing their daily meetings and add them into their google calendar. The success of this chatbot depends on the ability of the chatbot to understand input complex instruction from the user, extract correct information from it and add it to google calendar.

From 1966 onwards, business proprietors and developers understood the convenience chatbots can give to end clients, particularly when the data can be arranged into concrete and unsurprising subjects [4]. For instance, a repair store which has professionals who can repair fridges, microwaves and clothes washers could give its end clients a conversational experience with the clients in a human-like mold. At that point, since it is exceedingly likely that clients would refer to the thing they have at home which is broken, the chatbot could refer them to the fitting specialist with their separate telephone number, their address, and other significant pre-spared data open to the chatbot [5].

Influenced by the Turing Test idea [6], Joseph Weizenbaum, a German-American computer scientist and professor at MIT, published a program in 1966 called ELIZA [3]. ELIZA can identify keyword(s) in the user's input and after that create an answer containing this keyword. ELIZA should not be perceived as an "intelligent" program, but instead as a program which just gives the illusion of intelligence. ELIZA's essential rationale is to have the capacity to give reactions and advance the conversation in a human-like fashion. ELIZA breaks down the information given by the user via hunting down expressions and keywords, and gives pre-

1.2 Literature Review

AI can be considered as the most disruptive technology ever [1]. In the field of computer science, the implementation of intelligent agents (chatbots) that can have natural language conversation in a human way has been studied for a relatively long time [8]. The first chatbot called Verbot was created by Michael Loren Mauldin [2]. The **Verbot** (Verbal-Robot) was a popular chatterbot [5] program and Artificial Intelligence Software Development Kit (SDK) for the Windows platform and for the web.

From 1966 onwards, business proprietors and developers understood the convenience chatbots can give to end clients, particularly when the data can be arranged into concrete and unsurprising subjects [4]. For instance, a repair store which has professionals who can repair fridges, microwaves and clothes washers could give its end clients a conversational experience with the clients in a human-like mold. At that point, since it is exceedingly likely that clients would refer to the thing they have at home which is broken, the chatbot could refer them to the fitting specialist with their separate telephone number, their accessibility, and other significant pre-spared data open to the chatbot [5].

Influenced by the Turing Test Idea [6], Joseph Weizenbaum, a German-American computer scientist and professor at MIT, published a program in 1966 called ELIZA [3]. ELIZA can identify keyword(s) in the user's input and after that create an answer containing this keyword. ELIZA should not be perceived as an "intelligent" program, but instead as a program which just gives the illusion of intelligence. ELIZA's essential rationale is to have the capacity to give reactions and advance the conversation in a human-like fashion. ELIZA breaks down the information given by the user via hunting down expressions and keywords, and gives pre-

arranged pre-modified reactions. For example, given a sentence like "I used to hang out with my father a lot in my childhood.", ELIZA would behave as follows:

- Receive the input and store them in memory for further analysis
- Search the sentence, for already defined keywords
- For the sake of this example, we could think of the keywords being "mother", with a response like "tell me more about your mom" or "father", with a response like "tell me more about your Father." This follows from the following rules
- If any of the words within the sentence matches a keyword, then fire back the pre-programmed response to the user
- If there is no clear match then give back a default answer like "Sorry, don't know about that."

Given the above steps, an illusion of understanding is created even though the analysis method is trivial from a development perspective [3].

ALICE (Artificial Linguistic Internet Computer Entity) is another conversational chatbot inspired by ELIZA [6]. A.L.I.C.E. (Artificial Linguistic Internet Computer Entity), also known as Alicebot is a program that engages in a conversation with a human by applying some heuristic pattern matching rules to the human's input. ALICE won the Loebner prize (2003) competition three times in 2000, 2001, and 2004 - The Loebner competition is the way used nowadays to judge how much a chatbot could convince a user that it is a real human by chatting for 10 minutes.

Meekan is a scheduling assistant that matches everyone's calendars, and quickly finds common free times [2]. This agent integrates multiple calendar scheduling tasks. This scheduler

allows for the evaluation of time conflicts of multiple users at one time. Meekan tackles the largest scheduling conflicts for even the most hectic team tasks. Utterances that users can offer this scheduler include the following:

- *John:* "Meekan, schedule the project launcher next week and add a room."
- *Joe and I:* "Meekan, we want to have lunch at Joe's Pizza next week."
- *John:* "Meekan, ask everyone if they can delay tomorrow's meeting by five minutes."

Meekan offers inspiration to the current efforts in one major way, through the demonstration of multi-user task conflict management. Unlike this agent however, the conversational agent developed for this thesis aims to handle single-user tasks conflicts and scheduling challenges.

Another available agent like Self Planner can be considered as inspiration as well. Self-Planner is an intelligent web-based calendar application [10] that tries to accommodate the tasks in the user's calendar, considering a variety of constraints (e.g. location, preferences, periodicity of task, duration of tasks, and utility [amount of attention and effort required to complete a task]). This agent allows for the reservation of a specific block during pre-designated intervals. One of the most useful features of Self Planner is the fact that it considers temporal relationship between events. For example, if a user asks to schedule a meeting at a cafe at noon and give a lecture at the university at 2:00pm, the bot will consider how long it will take to get from the coffee to the university before scheduling the ending time for the meeting at the coffee shop, through the utilization of Google Maps. Another useful feature is the consideration of the periodicity of events. A user may make the following request: "Schedule a counseling appointment every Wednesday at 3:00pm," and the bot will automatically create an instance of

this event for each upcoming week. Such auto-instance creation can be allowed for daily, weekly, monthly, and yearly events.

Emma is a personalized calendar management assistant that aims at helping the user to schedule events like group meetings while considering the user's preferences [11]. Emma reads the calendars of all participants and generates a set of potential schedules for the meeting. Although Self Planner and Emma are intelligent, they were not implemented with the intention to have human-like emotions; the conversation with the bot does not feel like talking to a human.

Another area that chatbots have been found useful at is the education domain where the bots can answer questions that will help the teacher to see where students have problems and, what questions students ask. Log files could be generated by the chatbot and accessed by the teacher to gauge the student learning and help student's weaknesses. Sofia is a chatbot developed to help students with Mathematics [12]. Sofia can chat with users and at the same time chat with other mathematical agents such as Pari and Mathematica to help with solving Algebra problems. The "brain" of the bot contains text files that mainly focuses on Math and other common knowledge to make Sophia friendly to use. Sophia is trained to tell jokes and is familiar with movies in which Math plays a role.

Chatbot assistants have been also used in other domains like E-Commerce and business. Bots can be virtual assistants to customer service agents. They can help the customers with simple tasks like answering FAQ's, suggesting help articles and routing queries [27]. Other bots have been used in banking where they become the primary mode of interaction for the

customers. For example, Masterpass enabled bots by Mastercard creates an amazing shopping experience without leaving messenger [27].

1.3 Project Goal:

This thesis aims to develop a task-based conversational agent which has NLP capabilities that can help the user schedule meetings in real time while providing recommendations based on current weather and traffic information.

1.4 Thesis Organization:

This thesis is organized as follows:

Chapter 2 presents background of chatbot and recent trends in scheduling tools. It also discusses important steps in human-computer interaction, types of chatbot and different chatbot development platforms available in the market. The next chapter discusses the architecture of the system and talks about various components of the conversational agent. The last chapter discusses the agent's evaluation and the agent's strengths and limitations.

important life milestones or missing opportunities for career advancements. Not to mention, everyday life of the 21st-century modern adult can be plagued with an overabundance of information to attend to. Tools that support the maintenance of the responsibilities offer a

Chapter 2

Background

2.1 Introduction

Scheduling is the management of task ordering and resource distribution among those tasks. A scheduler is a person who carries out this process and creates the schedule, a basic time management tool consisting of chronologically ordered tasks. Schedules are usually organized into calendars, timetable-based systems used for organizing chronological social, religious, professional, and administrative events. A calendar consists of an ordered list of chronological events. Calendars and planners are ancient artifacts in human history. In fact, human beings have been using some form of calendars since prehistoric times. Celestial motions (e.g. sun and moon) served as the earliest forms of calendars. Hence planning is an aspect of human life that has greatly evolved over the past few centuries.

Most people could utilize some help with managing their lives. Task management presents a common challenge for many people in today's world. A world in which the average working professional is plagued with endless to do lists (tasks to remember, news updates to keep up with, ... etc.). Many are ridden with the need for reminders to make work meetings, keep up with recurring bills, show up to parent teacher meetings, recitals, and soccer practices. Forgetting a responsibility could mean missing a payment and incurring credit penalties or forfeiting the opportunities to be present in the lives of loved ones during some of the most

important life milestones or missing opportunities for career advancements. Not to mention, everyday life of the 21st century modern adult can be plagued with an overabundance of information to attend to. Tools that support the maintenance of the responsibilities offer a plethora of benefits for increasing productivity, saving time, and even decreasing anxiety related to task management. Chatbots have been used to address these challenges more commonly since circa 2010. We have seen this trend in several arenas, personal (e.g. home chatbots), professional (e.g. Clare.AI [13]), and academic (e.g. Ivy.ai). Hence productivity bots are becoming more and more common as well. These productive agents are often used for the management of scheduling tasks [12].

2.2 Recent Trends in the Development of Scheduling Tools

As previously mentioned, this thesis proposes the development of a conversational agent called Schedulio that helps with scheduling new tasks. Schedulio will be able to do the following:

- Add tasks and appointments through the Google Calendar API.
- Offer suggestions based upon external location-based factors (i.e. weather, temporal distance of scheduled events).

Tools such as search engines and personal assistants lack personality, evidence of their mechanical acceptance of input and production of output, and practicality, demonstrated by their intake of a single type of input, like keywords [28]. To address these issues, digital

assistant systems that combine the power of NLP, intelligent document categorization and retrieval systems were developed. Google assistants, Siri and Cortana are few examples of digital assistants which can help the user with almost anything.

In recent years, dialogue based, and conversational agents have become popular subcomponents of some of the most popular applications and home IoT technologies (e.g. Google home, Amazon Alexa, Facebook messenger, and IBM Watson business services). The inclusion of natural language within such tools is becoming a trendy request among modern-day consumers.

Planning is most effective through verbal utterances. For example, most people tend to go through their tasks by saying them each morning when they wake up or throughout the day as they think of them, (e.g. "I will take the dog out for a walk at 3:00pm today, after I come home from work"). In recent years, app development for productivity tools have taken off. Apps like Google Calendar, Boomerang, QuickCal, and ToDoist have made planning tasks easier for many. However, many of these are lacking the dialogue interfacing component. Our agent developed for this thesis, Schedulio, will utilize these two major benefits: 1. the prevalence of natural language processing algorithms within modern technologies, 2. human-based tendency to plan through verbal utterances.

Ambiguity is one of the major reasons why natural language processing development has lagged. Another big problem is language variability. There can be various ways of expressing a certain meaning. The third problem can be word sense disambiguation. One word can mean different meaning when written in different contexts [29].

While the development of natural language understanding (NLU) computerized tasks and technologies have lagged other developmental milestones (i.e. internet of things), most users still prefer to interact with scheduling APIs through natural language. In the past, the Google Calendar app utilized the "quick add" feature, with which users could add events to

their calendars using brief natural language phrases within the task bar (e.g. schedule an appointment with the ophthalmologist at 8am next Tuesday). However, this text-based feature was removed from the recently updated version of the app. Most consumers have requested that the feature be re-included in future releases of the application. This is only one example of the new trend for the popular demand to incorporate dialogue based agents within daily task management apps.

Schedulio is responding to this development trend through the development of an agent that takes advantage of dialogue-based interfaces.

2.3 Natural Language Processing (NLP) and Natural Language Understanding (NLU)

Natural language is a fundamental part of conversational agents. Artificial intelligent chatbots are harnessed as a part of businesses, for example, Banking frameworks, Client administrations, Education, Scheduling [16]. From a conceptual point of view, the important technologies behind conversational agents are Natural Language Processing and Natural Language understanding. Natural Language Processing (NLP) sits at the intersection of computer science, artificial intelligence, and computational linguistics [17]. Natural language Processing consists of different techniques for interpreting human language, ranging from statistical and machine learning methods to rules-based and algorithmic approaches [3]. Basic NLP tasks include tokenization and parsing, lemmatization/stemming, part-of-speech tagging, language detection and identification of semantic relationship [4]. All the above terms can be explained briefly as follows:

- Lexical Analysis – involves identifying and analyzing the structure of words.

Lexicon of a language means the collection of words and phrases in a language. Lexical analysis is dividing the whole chunk of text into paragraphs, sentences, and words.

- Syntactic Analysis (Parsing) – includes analysis of words in the sentence for

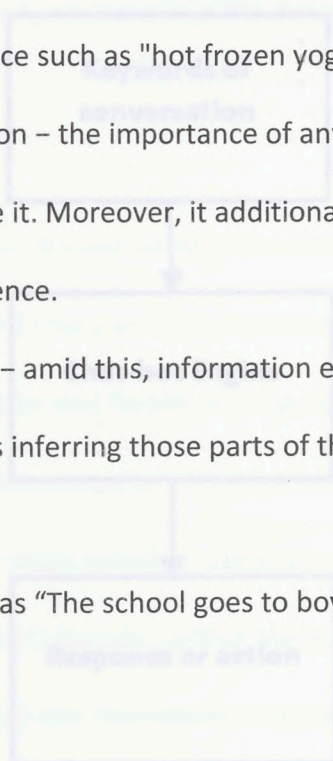
language structure and arranging words in a way that demonstrates the relationship between the words.

- Semantic Analysis – draws the correct importance or the word reference significance from the content. The content is checked for meaningfulness. It is finished by mapping syntactic structures and questions in the undertaking space. The semantic analyzer disregards sentence such as "hot frozen yogurt".

- Discourse Integration – the importance of any sentence relies on the significance of the sentence just before it. Moreover, it additionally achieves the importance of promptly succeeding sentence.

- Pragmatic Analysis – amid this, information exchanged is re-translated on what it really implied. It includes inferring those parts of the language which require real-world knowledge.

- The sentence such as "The school goes to boy" is rejected by English syntactic analyzer.



Natural Language Understanding (NLU) is a subset of NLP that deals with the much smaller, however similarly imperative aspect of how to best deal with unstructured data sources and change over them into an organized frame that a machine can comprehend and follow up on. While people can easily deal with errors, swapped words, withdrawals,

expressions, and different peculiarities, machines are less proficient at taking care of capricious sources of info [18].

2.4 Human-Computer Interaction Using Natural Language Dialogue

Humans and Computers can interact in natural language in two ways. The first way is by typing the keywords, (b) pattern matching techniques which are used in Chatbot engines, and (c) text and the second is by voice dialogue. Chatbots have become one of the best ways for Human-Computer Interaction. The following diagram in Fig. 1 shows the important stages of human computer conversation:

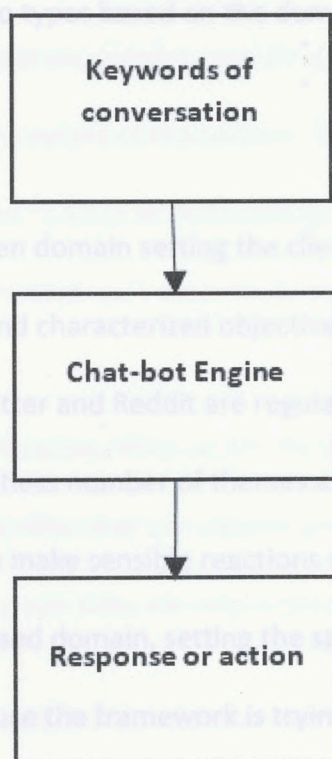


Fig. 1 Important stages of human-computer interaction

The factors that affect human computer interaction quality in chatbot conversational systems are: (a) the procedures used to analyze the text using different grammar sets in order

to develop the keywords, (b) the pattern matching techniques of the chatbot engine, and (c) the kind of response with respect to the specific application [19].

The factors affecting computer interaction in chatbot conversational systems are as follows: (a) methods used for analyzing text using different grammar sets in order to develop the keywords, (b) pattern matching techniques which are used in Chatbot engines, and (c) response as per the specific application.

2.4.1 Types of Chatbots

Chatbots can be classified into two types based on the domain:

(1) Open Domain Chatbots

(2) Closed Domain Chatbots

Open Domain Chatbots: In an open domain setting the client can take conversation anyplace.

There isn't really have an all-around characterized objective or goal. Discussions via web-based

networking media locales like Twitter and Reddit are regularly open domain — they can go into a wide range of bearings. The limitless number of themes and the way that a specific measure of world knowledge is required to make sensible reactions make this a difficult issue.

Closed Domain Chatbots: In a closed domain, setting the space of possible inputs and outputs

is to somewhat constrained because the framework is trying to accomplish a certain objective.

Specialized Customer Support or Shopping Assistants are cases of closed domain bots. These

systems don't should have the capacity to mimic humans or tend to behave in a human way;

they simply need to satisfy their particular undertaking as productively as could reasonably be

expected. Without a doubt, clients can even now take the discussion anyplace they need,

however, the framework isn't required to deal with all these cases — and the clients don't anticipate that it will [20].

2.4.2 Chatbot Development Platforms

There are different types of platforms which are available for making Intelligent chatbots. All these platforms have different complexity levels, expressive powers, and integration capabilities. Depending on the chatbots needs, some platforms are more appropriate than others. So how does a chatbot understand what the user is trying to say? The two main techniques to understand this are: Pattern Matching and Intent Classification.

Pattern Matching: In Computer Science, pattern matching is the act of checking given sequence of tokens for the presence of some values of the pattern. The pattern has to be exact for pattern matching [21] For example, “I want to fly to Atlanta, Georgia from Columbus, Ohio ,on December 31” can match a pattern such as “I want to fly to <CITY> from <CITY> on <DATE>” [22].

Intent Classification: Intent Classification relies on Machine Learning Techniques. A set of examples are a must to train a classifier that can choose given a user input among all possible intents. In the example above, city and date are very important factors.

There are many platforms which help with developing intelligent agents. Below are some development platforms which are discussed in detail:

Dialog Flow(API.AI):

Dialog Flow is a platform used for building intelligent chatbots using NLU. Intents and contexts are used for modelling a behavior of chat bots. In simple terms, the intention of the users when they are interacting with the bot. Context is equally important for the chatbot to understand what the user is trying to say. We use context for saving the users location and other values that persist throughout the conversation. For example, if the user is looking for a restaurant. The bot might ask if the user want's a specific type of food. The user can then provide a food type or ask for a suggestion — in both cases, the data in the first input, the type of venue (restaurant), is saved as context [23]. At the point when a user inputs information in Dialogflow.com platform, it is first checked on the off chance that it coordinates a pre-characterized intent. Dialogflow.com has an element named "Default Fallback expectation" to deal with the user inputs that don't coordinate any pre-characterized goal [23]. The agent can be trained using various natural language examples. Dialog flow uses pattern-matching to extract relevant entities and events [7].

Entities:

The user can create custom entities or use proposed entities by Dialog flow. example, if the user is trying to order a pizza through the bot then type and size of pizza are user-defined entities, while the address would be system defined quantities.

Slot-filling capabilities:

Slot-filing is the one of the major advantage of Dialog flow, it brings flexibility and power. Using slot-filing, the user can define the fields that are mandatory and the fields that are optional. The user does not have to worry about missing information with Dialog flow. Dialog flow will ask for each mandatory field until they are filed by the user.

Server side coding: Web hook combination: it passes data for every "Bot sends" order into a web

Using web hook integration in Dialog flow, custom coding on the server side becomes very simple. Dialog flow passes information to a matched intent into a web service and gets result from it. It is possible to change the chatbot side logic to some extent. The developer can decide which intents can call web hook and whether the web hook is going to be called or not during the slot-filing process.

Wit.ai: It ai, the user can define user defined entities or use pre-built ones.

In Wit.ai the important concept to model a chatbot are stories. Each story is depiction of a conversation flow. The NLP engine of Wit.ai is trained with examples. The intent is not mandatory as it is a user-intent. If a complex chatbot must be created using this platform, the platform can group many intents in stories. When a use makes similar request, the entities are extracted, and the request is processed as per the logic is defined by the developer.

A story can be viewed as a graph of user intents. You can include branches that are activated by conditions, for example, the presence or not of the variable values, that are extracted from the user input. This enables the user to characterize a conversation flow. In addition, the user has a bookmark component, used to hop amongst intents and furthermore between stories.

To connect to the server side, you have "Bot sends" summons, that fundamentally calls to functions. An exceptionally intriguing point is that you can set the role of the entities in a phrase. For instance, in "I need to travel to Columbus, Ohio from Bangkok, Thailand, on March 31", you can express that the main city is the takeoff and the second one the goal.

Server-side coding: Server-side coding is contained in local systems such as standalone servers, but in cloud-based management. This cloud-based management can include options for servers, infrastructure,

Wit.ai proposes a web hook combination: it passes data for every "Bot sends" order into a web service and gets a result from it. On the server side, you are normally going to make or grow the context of the conversation. The outcome sent to Wit.ai can include, adjust and erase setting variables utilized on the chatbot side.

Entities: All messaging, storage, computing, game development, database and analytics. The

With Wit.ai, the user can define user defined entities or use pre- built ones.

2.5 Web Services

Web services are open standard (XML, SOAP, HTTP, etc.) based web applications that interact with other web applications for the purpose of exchanging data. Web services can convert your existing applications into web applications. A web service can also be defined as the collection of open protocols and standards which are used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services as an exchange system for data over computer networks like the Internet in a manner like inter-process communication on a single

computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

Amazon Web Services(AWS): send income and expense data to Amazon Simple Storage Service

In the present-day computing environment, much of network management and operation is no longer contained in local systems such as standalone servers, but in cloud-based management. This cloud-based management can include options for servers, infrastructure,

data analytics, security, and more. Cloud service providers are companies that provide different services that users can choose from to meet their needs. One such cloud service provider is Amazon Web Services ("AWS"). AWS offers many different services to users from virtual instances of servers and databases to machine learning technologies and even complete system monitoring. AWS is a cloud service platform and it has tens of services and solutions for topics such as AI, messaging, storage, computing, game development, database and analytics. The tools that were used to build the chatbots are the following:

AWS Lambda:

When using AWS Lambda, the developer can run code for without provisioning or managing servers. With Lambda, the user can run the code virtually any type of application or backend service without any administration. The code can be set-up to trigger automatically or to call it automatically from mobile apps, HTTP endpoints or in-app activity. AWS supports code written in a variety of languages like Node.js, Python and C#.

API Gateway:

API Gateway is used to for incoming requests for other AWS resources like AWS Lambda. It has RESTful (APIs) application programming interfaces for mobile and web applications to access to web services. For example, an application can call an API in API Gateway to upload a user's annual income and expense data to Amazon Simple Storage Service or Amazon DynamoDB, process the data in AWS Lambda to compute tax owed, and file a tax return via the IRS website. An app or client application gains access to AWS services through API gateway. In API Gateway, the frontend is encapsulated by method requests and method

responses, and the backend is encapsulated by integration requests and integration responses[29].

DynamoDB

DynamoDB is a NoSQL database that backs both document and key-value store models [3]. It guarantees a cluster of features, for example, seamless scaling, high availability, secondary, free-text search, strong consistency, and cross-region replication. It's anything but difficult to set up and Amazon gives a web interface to browse the tables in the database.

Slack:

Slack is an instant messaging and collaboration platform, see Fig. 2. It is meant for teams and workplaces. The user can chat one- to one as well as in groups. Slack allows the user to share all types of files, even those stored out of the solution itself. Slack also has free to use forever pricing tier.

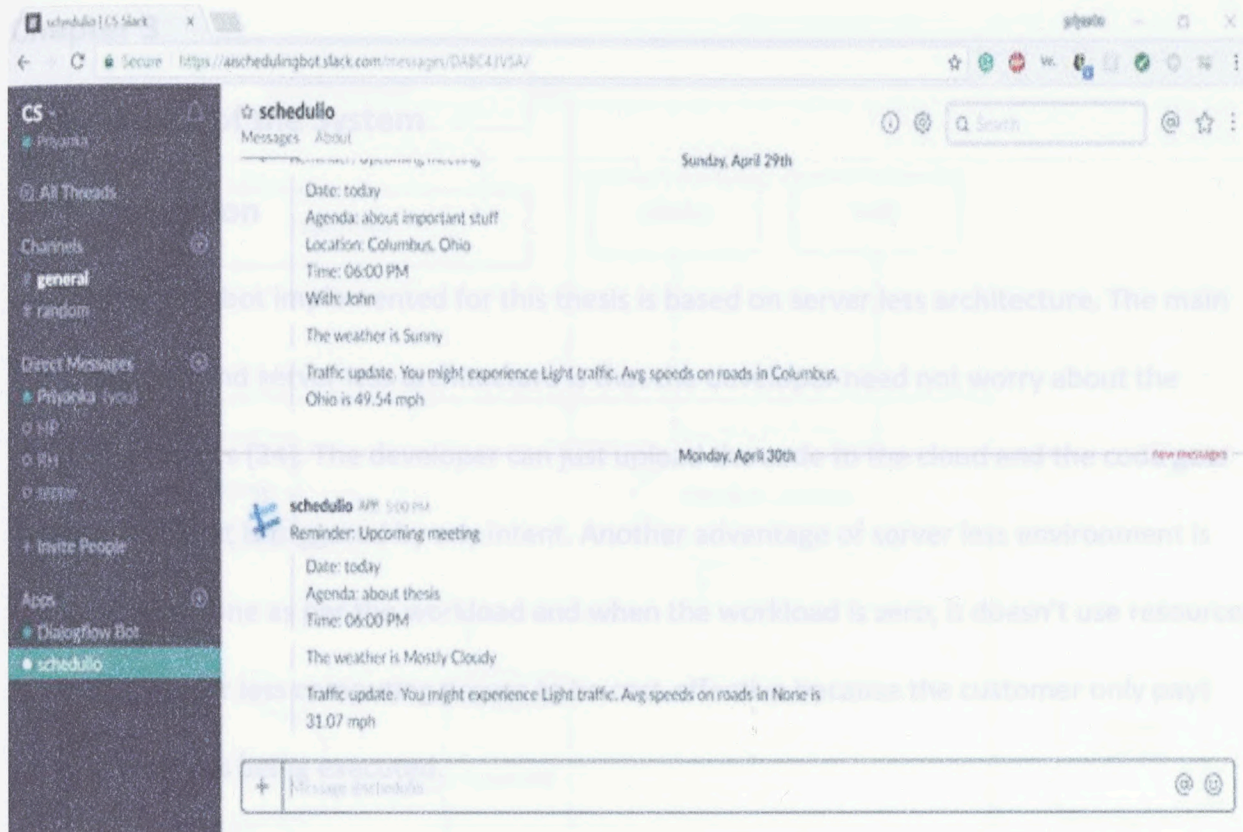


Fig. 2 Slack platform

Dialog flow:

Dialog flow is a framework for chatbot development. Dialog flow is owned by Google and has one of the best natural language processing capabilities. Dialog flow works with entities and intents. For example, in this project confirm schedule, cancel schedule etc are intents while agenda for the meeting can be considered as an entity.

The Intents section consists of four sections:

- Training phrases
- Action
- Response
- Contexts

Chapter 3

Architecture of the System

3.1 Introduction

The Chatbot implemented for this thesis is based on server less architecture. The main motivation behind server less architecture is that the developer need not worry about the managing servers [24]. The developer can just upload the code to the cloud and the code gets executed when it is triggered by any intent. Another advantage of server less environment is that scaling is done as per the workload and when the workload is zero, it doesn't use resources at all [25]. Server less computing proves to be cost-effective because the customer only pays when the code is being executed.

3.2 System Architecture

The architecture of Schedulio is cloud based architecture that is scalable, easy to manage and deploy, see Fig. 3. The code on the server run only when it is triggered by the agent. The agent is deployed on slack. The agent also gets the weather and traffic data and sends a notification to the user an hour before the meeting about the estimated traffic in the area and weather information.

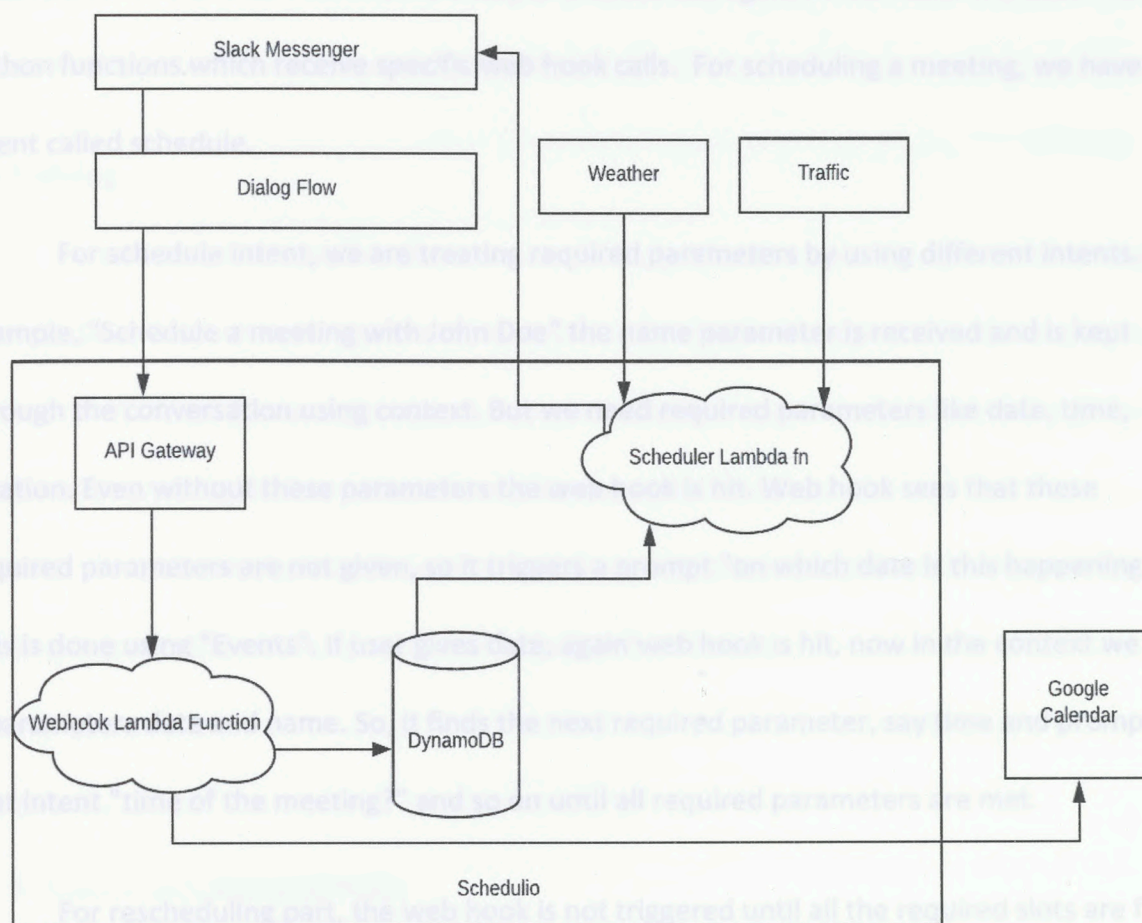


Fig. 3 Schedulio Architecture

Slack is the best platform for professional collaboration. Hence, slack was chosen as the chat platform. The conversational agent created in Dialog flow agent is deployed on Slack platform. The agent build on Dialogflow has options to deploy the agent on 14 different platforms. Dialogflow agent has intents and entities and events and context. Contexts help the agent to understand about the context of the user input. Whenever a user enters any command into Slack. It is sent to the Dialogflow agent. The conversational agent will identify the intents and entities from the user input and extract useful information from it. Based on the user input, intents will be triggered in the Dialogflow agent. Each intent in Dialogflow has a web

hook attached to it. This web hook is a way to connect the agent to the code. The code has python functions which receive specific web hook calls. For scheduling a meeting, we have an intent called schedule.

For schedule intent, we are treating required parameters by using different intents. For example, "Schedule a meeting with John Doe" the name parameter is received and is kept through the conversation using context. But we need required parameters like date, time, location. Even without these parameters the web hook is hit. Web hook sees that these required parameters are not given, so it triggers a prompt "on which date is this happening?", This is done using "Events". If user gives date, again web hook is hit, now in the context we have 2 parameters date and name. So, it finds the next required parameter, say time and prompts that intent "time of the meeting?" and so on until all required parameters are met.

For rescheduling part, the web hook is not triggered until all the required slots are filled. Once all the required parameters are in the context, a new item is added to DynamoDB in the table "schedule" and parallelly the Google calendar add event is triggered. For adding an event to the google calendar, the server looks for users Google calendar access token from another DynamoDB table called "calendar".

For adding an event to the google calendar, the server looks for users Google calendar access token from another DynamoDB table called "Calendar". If the access token is not found, then the sign in using Google link is sent. Upon sign in, the parameters received earlier like date, time, location and other optional parameters are sent to an authentication handler web hook. But if the user has already logged in once, the access token would be already present. If

the access token was found, then the Google Calendar would have been created/updated/deleted in the intent handler web hook itself.

3.3 Training

To achieve good classification accuracy, it is important to train the agent with enough training data. When the user saves an intent, Dialogflow trains the agent with the new data that the user just added. The updates can be seen only after the agent has finished training. The more training examples user adds, the smarter the agent becomes.

The user can train the agent in Dialogflow to help it understand what the user says, see Fig. 4.

The training phrases can be typed in the “User says” section.

The screenshot shows the 'Training phrases' section of the Dialogflow console. At the top, there is a search bar labeled 'Search training phrases'. Below it, a list of training phrases is displayed, each with a delete icon (X) on the right. The phrases are:

- Add user expression
- Schedule a meeting in half an hour from now
- Schedule my meeting with Dr Hodhod today
- I have a meeting at 2pm today
- Add a meeting for me
- I have a meeting with John Smith tomorrow at 8pm in London

The phrase 'I have a meeting with John Smith tomorrow at 8pm in London' is selected, and its resolved parameters are shown in a table below:

| PARAMETER NAME | ENTITY | RESOLVED VALUE | |
|----------------|-----------------|----------------|---|
| given-name | @sys.given-name | John | X |
| last-name | @sys.last-name | Smith | X |
| date | @sys.date | tomorrow | X |
| time | @sys.time | 8pm | X |
| location | @sys.location | London | X |

Fig. 4 Training phase of Dialogflow Agent

Annotation is the process of linking a phrase or word to an entity. When the user adds training phrases, they are annotated automatically. A user can edit the linked entity and the parameter name assigned to it in either the review window that opens when you click on the annotated example, or the parameter table of the 'Action' section.

3.3.1 Actions

When a specific intent is triggered by user's input, the agent should perform some step. This step is known as Action. Actions have parameters to extract information from user input. Parameters can be filled in automatically from the training data examples or added manually. Clicking on an annotated word in an example will reveal a table with data on the chosen entity, see Fig. 5. Both action name and its parameters are defined in the Action section of an intent. The actions can be shown in the JSON format as follows:

```
{"action": "action_name"}
```

```
{"parameter_name": "parameter_value"}
```

Action & parameters ?

Enter action name

| REQUIRED ? | PARAMETER NAME ? | ENTITY ? | VALUE | IS LIST ? | PROMPTS ? |
|-------------------------------------|------------------|---------------|-------------|--------------------------|--------------------|
| <input type="checkbox"/> | date | @sys.date | Sdate | <input type="checkbox"/> | --- |
| <input checked="" type="checkbox"/> | geo-city | @sys.geo-city | Sgeo-city | <input type="checkbox"/> | For what city d... |
| <input type="checkbox"/> | Enter name | Enter entity | Enter value | <input type="checkbox"/> | ... |

+ New parameter

Fig. 5 Actions and Parameters

3.3.2 Response

The user can improve your agent eloquence by adding multiple variations in the text response per intent. When the same intent has been triggered more than once, different text response variations will be unrepeatable until all options have been used. It'll help make agent's speech more human-like.

3.3.3 Contexts

Contexts represent the current context of a user's request. This is helpful for differentiating phrases which may be vague or have different meanings depending on the user's preferences, geographic location, the current page in an app, or the topic of conversation. For example, if a user is listening to music and finds a band that catches their interest, they might say something like: "I want to hear more of them". As a developer, you can include the name of the band in the context with the request, so that the agent can use it in other intents.

3.4 External Apps

3.4.1 HERE Predictive Traffic

HERE Technologies understands the negative effect of traffic congestion; it is an issue that is difficult to forecast and subject to change considering various components. HERE Predictive Traffic was intended to take care of these issues [26].

HERE Predictive Traffic uses more than one trillion GPS data points and precisely estimates to what extent the trek will take will take by figuring real-time traffic, historic traffic flow data and different elements like normal seasonal conditions and holidays. This project uses

the HERE predictive traffic API to predict the future traffic congestion at a given time in any and the average speed of traffic at a given point of time in any city.

System Evaluation and Results

3.4.2 Yahoo Weather App

4.1 Introduction

We are using Yahoo weather API for this project. Yahoo gives up-to date weather information for any location at any time of the day. We will use this feature to notify the user about the weather during the time of the meeting.

Task completion success: Task success can be measured by evaluating the correctness of the total solution. For a frame-based architecture, this might be the percentage of slots that were filled with the correct values or the percentage of slots that were completed.

4.2 Evaluation Metrics for Schedulo

Schedulo is a task based chatbot. It is a calendar assistant which helps schedule meetings and helps the user with weather at the time of the meeting and estimated traffic around that time.

We are going to evaluate Schedulo based on two factors:

1. Slot Error rate of a sentence
2. End-to-End Evaluation (Task Success)

Slot Error rate for a sentence: Slot Error can be defined as errors of the chatbots in fulfilling slots. For Example, if the user says, "Schedule a meeting with Mr. John Smith at 2 pm on Tuesday". Table 4.2.1 shows how the chatbot parses what the user said to fill the slots.

Chapter 4

System Evaluation and Results

4.1 Introduction

Evaluation is crucial in dialog systems. As the agent built for this project is task-based we can measure the task success for the agent. Task-oriented dialog agents are designed for a task and set up to have short conversations. The success of task base dialogue agents depends on task completion success. Task completion success can be defined as follows:

Task completion success: Task success can be measured by evaluating the correctness of the total solution. For a frame-based architecture, this might be the percentage of slots that were filled with the correct values or the percentage of subtasks that were completed.

4.2 Evaluation Metrics for Schedulio

Schedulio is a task based chatbot. It is a calendar chatbot which helps schedule meetings and helps the user with weather at the time of the meeting and estimated traffic around that time.

We are going to evaluate Schedulio based on two factors:

1. Slot Error rate of a sentence
2. End-to End Evaluation (Task Success)

Slot Error rate for a sentence: Slot Error can be defined as errors of the chatbots in fulfilling slots. For Example, if the user says, "Schedule a meeting with Mr. John Smith at 2 pm on Tuesday". Table 4.2.1 shows how the chatbot parses what the user said to fill the slots.

Table 4.2.1 – Slot Filling

| | |
|--------|------------|
| Slot | Fillers |
| Person | John Smith |
| Time | 2 am |
| Day | Tuesday |

Slot error rate: 1/3

Task success: At end, was the correct meeting added to the calendar?

4.2.1 Failure cases in Schedulio

In Triggering Schedule intent:

Success: words like "add, schedule, note down" combined with contextual words like "meeting, interview, conference" works well.

Failure: couldn't detect any.

One-line scheduling

Failure:

1. "Schedule a meeting in London on the 7th at 4:30pm with Dr. Abdul Kalam"

Remarks:

- a. Asks for agenda
- b. Does NOT pick up the name "Dr. Abdul Kalam"

Success:

2. "Schedule a meeting in London on the 7th at 4:30pm with John Doe"

Remarks: no meeting

a. Asks for agenda (at time is the meeting happening?)

b. picks up the name "John"

Conclusion: Can be trained better to pick more names. Could be issue with Google's side or with limits of our input dataset

Agenda

What is the agenda? or What's the meeting about?

Success:

1. about resolving internal conflicts
2. about improving performance
3. about empowering women
4. about making website design better

Failure:

1. about world peace
2. need to talk to the sales team and set correct deadlines (is up for most cases)
3. to make sure that the next version to be released is free of the major bugs
4. chat with developers on the right tech stack

Conclusion:

1. needs the trigger word "about" before the actual agenda to increase chances of pickup
2. Not all agenda are picked up properly. (not leap year, as picked up Feb 29, 2020 even though we

did not give any year information)

Time of the meeting

When asked "At what time is the meeting happening?"

Failure: *gust*

1. 6oclock

2. six o clock *1 month*

Success: *on: slim chance of user giving bad information, picks up for most cases.*

1. 6 oclock

2. 3 *on asked "Where is this happening?"*

3. 2 AM

4. six pm *uth Market, Kowai Nagar (c), Bangalore (not picked up)*

5. nine 30 *are, hosth*

6. 4 a.m.

7. 17hrs *on national park, south sudan (picked up as just "park")*

8. 12 twenty

Conclusion: Slim chance of user giving bad information, picks up for most cases.

Date of the meeting *Drive, Las Vegas*

When asked "On which date is this happening?" *aria*

Failure: *rovo*

1. mid of next month *for some locations from Africa taken from google maps (dialogflow being*

2. Feb 30 (special case: resolves to the next leap year, so picks up Feb 29, 2020 even though we did not give any year information)

Success:

1. 2018-11-12

2. 11 august

Screenshots of the System on Slack Platform

3. 12 sep

4. first of next month

Conclusion: slim chance of user giving bad information, picks up for most cases.

Location

When asked "Where is this happening?"

Failure:

1. 60, South Market, Kidwai Nagar (e), Bangalore (not picked up)

2. 41 Fox Lane, boath

3. banglai

4. Radom national park, south sudan (picked up as just "park")

Success:

1. 33 St Denys Road, Prescott

2. 533 Hickory Ridge Drive, Las Vegas

3. 4764 Coal Street, seven springs, PA, Pennsylvania

4. Kemerovo

Conclusion: Fails even for some locations from Africa taken from google maps (dialogflow being a google service). Not the most accurate slot, 50-50 failure from testing.

4.2.2 System Run:

Screenshots of the System on Slack Platform:

Fig. 6 shows a screenshot that the bot can have small conversations with the user. Dialog Flow's small talk feature helps it to have small conversations.

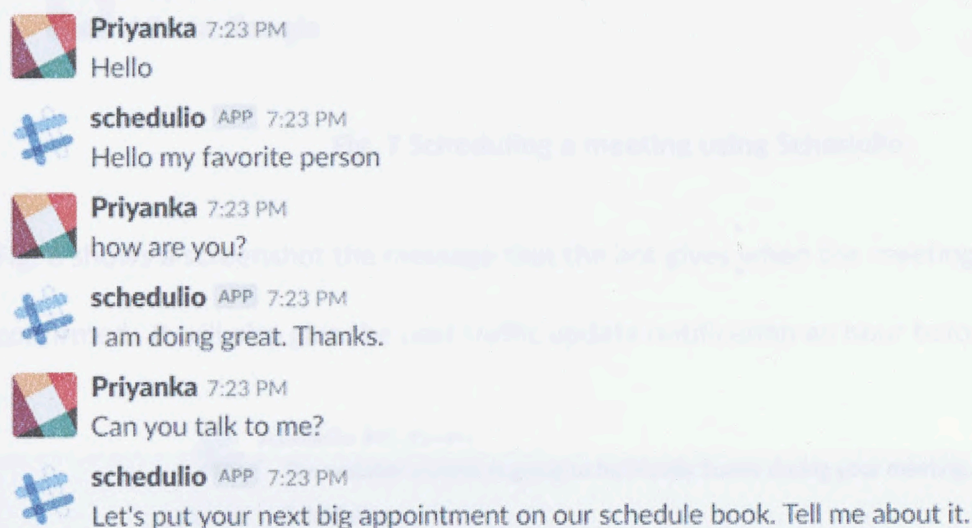




Fig. 6 Small Talk with Dialog flow agent


Fig. 7 shows a screenshot that the bot can understand what the user is trying to say. It will extract meaningful information from what the user said and ask questions if it is missing anything.

 **schedulio** APP 7:23 PM


Let's put your next big appointment on our schedule book. Tell me about it.

 **Priyanka** 7:24 PM


Add a meeting with Mr. John on Monday at 8 am

 **schedulio** APP 7:24 PM


What's the agenda?

 **Priyanka** 7:24 PM

about research

 **schedulio** APP 7:24 PM


Where is this happening?

 **Priyanka** 7:24 PM

Atlanta, Georgia

Fig. 7 Scheduling a meeting using Schedulio

Fig. 8 shows a screenshot the message that the bot gives when the meeting schedule is confirmed. It will also give the user traffic update notification an hour before the meeting.

 **schedulio** APP 7:24 PM

The weather outside is going to be Mostly Sunny during your meeting.

7:24 PM **Scheduling**

Date: tomorrow
 Agenda: about research
 Location: Atlanta, Georgia
 Time: 08:00 AM (3 kB) ▾

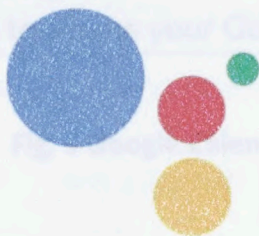


Fig. 8 Confirmation Notification from Schedulio

There are two options below this meeting schedule, see Fig. 9. "Confirm Schedule" and "Cancel". Once the user confirms the schedule it will add the meeting to the users google Calendar. If the user is using this Slackbot for the first time then it will ask the user to sign in into his google calendar.

Time: 08:00 AM (3 kB) ▾

Once the user signs in, the bot would not ask the user to sign in again. From next time, the meeting will be directly added to the user's calendar as shown in Fig. 11 below.

Google Calendar Sign In
Allow Schedulio to access
your Google Calendars

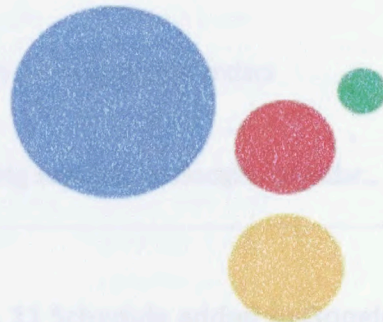


Fig. 11 Schedule added to Google Calendar

Confirm Schedule

Cancel

Google Calendar Sign In

Allow Schedulio to access your Google Calendars

Fig. 9 Google Calendar Sign-in

| User Input | Human extracted | Bot extracted | Slot Error | Task Completed | Remarks |
|----------------------|-------------------------------|-------------------------------|------------|----------------|---------|
| Could you schedule a | Person Name: [Name] [Time] | Person Name: [Name] [Time] | 0 | Yes | |

Once the user signs in using his credentials. The message in Fig.10 will be shown.

| | | | | |
|---|--------------------|--------------------|---|-----|
| Google Calendar Sign In | Date: | | | |
| Allow Schedulio to access your Google Calendars | | | | |
| Sign in with Google | | | | |
| Successfully signed in. Adding schedule to Google Calendar... | | | | |
| Help me meet with Hanna tomorrow at 3 pm | Person Name: Hanna | Person Name: Hanna | 0 | Yes |
| | Date: Tomorrow | Date: Tomorrow | | |
| | Time: 3 pm | Time: 3 pm | | |

Fig. 10 Google Calendar sign-in Confirmation

Once the user signs in, the bot would not ask the user to sign in again. From next time, the meeting will be directly added to the user's calendar as shown in Fig. 11 below.

| | | | | |
|---|-------------------------|-------------------------|-----|----|
| Google Calendar Sign In | Date: | | | |
| Allow Schedulio to access your Google Calendars | | | | |
| Sign in with Google | | | | |
| Successfully signed in. Adding schedule to Google Calendar... | | | | |
| Let's have a meeting with Dr. Hodhod next week | Person Name: Dr. Hodhod | Person Name: Dr. Hodhod | 1/3 | No |
| | Date: Anytime | Date: Did not | | |
| | Time: 11 am | Time: 11 am | | |

Fig. 11 Schedule added to Google Calendar

4.3 Results

Bots are the beginning of human computer interaction this project, we built a conversational agent that helps the user schedule and reschedule meetings, along with weather and traffic data. Table 4.3.1 shows the evaluation Metric for Schedulio.

Table 4.3.1: Evaluation Metric for Schedulio

| User Input | Human extracted Slots | Bot extracted Slots | Slot Error Rate | Task Completed (Yes/No) | Remarks |
|----------------------|-----------------------|---------------------|-----------------|-------------------------|---|
| Could you schedule a | Person Name: Liz | Person Name: Liz | 0 | Yes | The agent added the correct meeting to the calendar bot |

| | | | | | |
|---|---|---|-----|-----|--|
| meeting with Liz at 2 pm tomorrow? | Date: Tomorrow Time: 2 pm | Date: Tomorrow Time: 2 pm | | | misinterpreted the timing part. |
| Help me meet with Hanna tomorrow at 3 pm. | Person Name: Hanna Date: Tomorrow Time: 3 pm | Person Name: Hanna Date: Tomorrow Time: 3 pm | 0 | Yes | - |
| I want to meet with Jim on Thursday at 11 am. | Person Name: Jim Date: Thursday Time: 11 am | Person Name: Jim Date: Thursday Time: 11 am | 0 | Yes | - |
| Let's have a meeting with Dr. Hodhod at 11 am next week. | Person Name: Dr. Hodhod Time: 11 am Date: Anytime next week | Person Name: Dr. Hodhod Time: 11 am Date: Did not pick up time correctly. | 1/3 | No | The task gets added after we specify the date. |
| Schedule a meeting on July 5 at 10 a.m to discuss a grant extension with Dr. Summers. | Person Name: Dr. Summers Time: 10 am Date: July 5 Agenda: To discuss a grant extension | Person Name: Did not pick up Time: 10 am Date: July 5 Agenda: Did not pick up. | 2/4 | No | The agent did not pick up the name of the person and the agenda correctly. |
| I need to meet with Daniel at 4:30p on Monday to plan for robotics. | Person Name: Daniel Time: 4:30pm Date: Next Monday Agenda: To plan for robotics | Person Name: Daniel Time: 4:30 am Date: Next Monday Agenda: Did not pick up | 1/4 | Yes | - |
| I will be in Atlanta next | Date: Next Monday (June 1) | Date: Jun 01 Location: Atlanta | 1/4 | Yes | The agent added the correct meeting to the calendar but |

| | | | | | |
|---|--|--|-----|------|---|
| Friday 8 - 5p for a STEM meeting. | Location: Atlanta Time: 8:00 PM Agenda: About STEM | Time: 08:05 PM Agenda: About STEM | | | misinterpreted the timing part. |
| I will be teaching at Girls Inc Tuesday through Friday from 8 am to 4 pm. | Date: Tuesday (29 May) Time: 8 am to 4 pm Agenda: Teaching girls | - | 3/3 | No | The agent failed to understand this sentence completely. |
| Jun 11, 5 pm, about improving sales | Date: June 11 Time: 5 pm Agenda: Improving Sales | - | 3/3 | No | The agent failed to understand this sentence completely. |
| Can you schedule a meeting at tomorrow! at 12:45PM? | Date: Tomorrow Time: 12.45 PM | Date: Tomorrow Time: 12.45 PM | 0 | Yes. | The agent understands this sentence well and picks up all the slots correctly. |
| Schedule an appointment with Gen. Miller at tomorrow in the evening | Date: Tomorrow Time: Evening Person name: Gen Miller | Date: Tomorrow Time: Not picked up Person Name: Gen Miller | 1/3 | No | The task could have been added if the user had mentioned specific time instead of saying evening. |
| Meet frank at tomorrow!, tomorrow. | Person name: Frank Time: tomorrow | Person Name: not picked up | 1/2 | Yes | The agent added the meeting without adding the person's name. |
| Schedule a meeting in London on the | Person Name: John Location: London | Person Name: John Location: London | 0 | Yes | Can be trained better to pick more names. Could be issue with Google's |

| | | | | | |
|--|---|--|-----|-----|--|
| 7th at 4:30pm with John Doe | Time: 4:30 pm | Time: 4:30 pm | | | side or with limits of our input dataset |
| I have a meeting on the third Monday of next month. | Date: 18 June | Date: 18 June | 0 | Yes | The agent understood the phrase and added correct meeting to the calendar. |
| Remind me about a meeting on the third day of next month. | Date: 3rd June | Date: 3rd June | 0 | Yes | The agent understood the phrase and added correct meeting to the calendar. |
| Dinner with Nanglinchi at 7pm tomorrow | Agenda: Dinner Person Name: Naglinchi Time: 7 pm Date: Tomorrow | Agenda: Dinner Person Time: 7 pm Date: Tomorrow | 1/4 | Yes | Added the dinner without the person's name. |
| Go to office at 10:00AM tomorrow | Agenda: Go to office Time: 10 am Date: Tomorrow | Time: 10 am Date: Tomorrow | 1/3 | Yes | Asks for Agenda. |
| Clean office from 10:00AM to 11:45 AM tomorrow | Date: Today Agenda: Clean office Time: 10:00 AM | Date: Today Agenda: Clean office Time: 10:00 AM | 0 | Yes | Added correctly. |
| Coffee date at 8:00AM with Shuri | Date: Jun 26 Agenda: Coffee Time: 08:00 AM | Date: Jun 26 Agenda: Coffee Time: 08:00 AM | 1/4 | Yes | Added |

| | | | | | |
|---|--|--|-----|-----|--|
| | Person's Name: Shuri | Agenda: Doctors Appointment | | | |
| Video meeting with Book club tomorrow at 2:30PM | Date: Jun 08 Time: 02:30 PM Agenda: Video meeting with book club | Date: Jun 08 Time: 02:30 PM | 1/3 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Baking meeting every Thursday at 10:00AM | Date: Jun 01 Agenda: Baking Time: 10:00 AM | Date: Jun 01 Time: 10:00 AM Agenda: Missed | 1/3 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Meeting with Gloriana 7/5/18 at 2pm | Date: Jul 05 Time: 02:00 PM Person's Name: Gloriana | Date: Jul 05 Time: 02:00 PM | 1/3 | Yes | The agent added the meeting without adding the person's name. |
| Tutoring Session with Tanya at 5pm on Wednesday | Date: Wednesday Person Name- Tanya Agenda: Tutoring Session | Date: Wednesday Person Name- Tanya | 1/3 | Yes | The agent did not pick up the name of the person and the agenda correctly. |
| Breakfast with friends on next Friday at 6pm | Date: Next Friday Time: 6 pm Agenda: Breakfast with friends | Date: Next Friday Time: 6 pm | 1/3 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Doctors Appointment Next Tuesday at 1:22PM | Date: Next Tuesday Time: 1:22 pm | Date: Next Tuesday Time: 1:22 pm | 0 | Yes | Added correctly. |

| | | | | | |
|--|---|-----------------------------------|-----|-----|---|
| | Agenda: Doctors Appointment | Agenda: Doctors Appointment | | | |
| Take Nickelle to the Mall Thursday morning at 8:30AM | Person's Name: Nickelle Agenda: Going to the Mall Date: Thursday | Date: Thursday Time: 8:30 am | 2/3 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Play Soccer with Tony every Friday at 8pm | Agenda: Play Soccer Person's Name: Tony Time: 8 pm Date: Friday | Time: 8 pm Date: Friday | 2/4 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Meet Timothy for Tea at 9:30am Friday | Person's Name: Timothy Agenda: Tea Time: 9:30 am Date: Friday | Time: 9:30 am Date: Friday | 2/4 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Lunch meeting with Vanessa at 12pm | Agenda: Lunch Meeting Person's Name: Vanessa Time: 12 pm | Time: 12 pm | 1/3 | Yes | Added the meeting but did not pick up the agenda correctly. |
| Meet with Jonnette Monday at 1:11AM | Person Name: Jonnette Time: 1:11 AM | Time: 1:11 am | 1/2 | | Added the meeting but did not pick up the agenda correctly. |

Results and Discussion:

The goal of this thesis was to build a task based chatbot prototypically to understand what it takes to develop a chatbot that has natural language processing capabilities with real-time weather and traffic data support. In terms of the prototypical chatbot, the goal of this thesis is fully met. Considering the average of slot-error rate of the chatbot, the efficiency of this chatbot is 30%. During the development of Schedulio, however, it was found that due to lack of available datasets for train the DialogFlow agent, the agent fails to understand some phrases which convey the same meaning. Some of the examples of this can be seen in the table X. For example, the agent failed to extract the agenda from the phrase "I need to meet with Daniel at 4:30 p.m on Monday to plan for robotics". But once it is trained on this phrase, it extracts all the information accurately. Another example can be if the input for meeting time is given as 4:30p instead of 4:30 p.m. then the agent assumes it as 4:30 a.m. This shows the limitations of the agent to understand ambiguous phrases and times.

To train the agent, a large corpus of examples of conversation is needed large dataset which has millions of examples to reach a decent quality of Natural language understanding. The task based conversational agent has been trained on a very limited number of examples which can be seen in Appendix A.

Based on the evaluation system (Slot error rate and task completion) the agent picks up the user utterances that are syntactically correct and specific and unambiguous. For example, the agent understands better if the user says, "I have a meeting at 5pm on Thursday" rather than "I have a meeting next week". The second phrase is more ambiguous as it does not specify the exact day and time. The agent successfully adds the task to the users google calendar and

helps the user with the weather information during the time of the meeting and sends the traffic update and reminder notification an hour before the starting time.

Future research:

The results from this work are promising and reflects the fact that a simple implementation of task based chatbot build using Dialogflow can be improved if trained large enough number of examples and can be improved over time.

A suggestion for the future research would be to use Common Sense Knowledgebases such as ConceptNet so improve the efficiency of the natural language processing component of the conversational agent.

Chapter 5: Appendices

Appendix A

| List of Intents | Training Phrases |
|-------------------------|---|
| Cancel_schedule | Cancel Cancel Schedule |
| Confirm_reschedule | Confirm Reschedule |
| Confirm_schedule | Confirm Schedule |
| Default Fallback Intent | I didn't get that. Can you say it again? I missed what you said. Say it again? Sorry, could you say that again? Can you say that again? One more time? What was that? I missed that. |
| Default Welcome Intent | Hi! Hello! Good day! Greetings! |
| Get_agenda | The meeting is about improving sales about cat training about launching a rocket I am going on a date working on design documents about a travel tour going fishing going to a movie about saving the world its about a get together with friends about launching a spacecraft about improving sales about project discussion its about building a survival kit for mars about saving the planet running in a marathon going to buy a new car going hiking about improving sales and marketing catch a flight chat with the CEO about creating large a hadron collider |

| | |
|--------------|---|
| | <p>its about writing a paper for the upcoming journal submission</p> <p>its a chat with the CEO</p> <p>about launching an aircraft</p> <p>about travelling</p> <p>going swimming</p> <p>about thesis</p> <p>its about how to redesign the app</p> <p>going camping</p> <p>about launching spacecraft</p> <p>project discussion</p> <p>I am going to court</p> <p>going shopping</p> |
| Get_meetings | <p>about how to make the code faster</p> <p>we are meeting for a cricket math</p> <p>meetup for football match</p> <p>agenda is about travelling</p> |
| Get_date | <p>next tuesday</p> <p>on may 17</p> <p>day after tomorrow</p> <p>coming friday</p> <p>24th may</p> <p>next thursday</p> <p>sunday</p> <p>on 11th of this month</p> |
| Get_location | <p>Orlando</p> <p>at chennai</p> <p>studio</p> <p>in Cincinnati</p> <p>italy</p> <p>chennai</p> <p>in Marylebone, London NW1 6XE, UK</p> <p>in Baker st</p> <p>West Fargo</p> <p>at paris</p> <p>at the meeting room</p> <p>rome</p> <p>123 6th St. Melbourne</p> |
| Reschedule | <p>london</p> <p>897 Sugar Ave. Annapolis, MD 21401</p> <p>at bangkok</p> <p>new mexico</p> <p>West Chicago</p> <p>paris</p> <p>West North Dr. Marlborough</p> |

| | |
|--------------|--|
| Schedule | <p>Honolulu</p> <p>in the office</p> <p>at 221b Baker St, Marylebone, London NW1 6XE, UK</p> <p>Deerfield Street Glenview</p> <p>at berlin</p> <p>at Baker st, Marylebone</p> <p>Pilgrim Avenue</p> <p>in space</p> <p>at the coffee shop</p> <p>coffee shop</p> <p>Bowman St. South Windsor</p> <p>atlanta</p> |
| Get_meetings | <p>pull up my reminders</p> <p>what's in my schedule?</p> <p>get reminders</p> <p>get meetings for this week</p> <p>go to reminders</p> <p>show me the reminders</p> <p>get meetings for today</p> <p>what are my meetings for tomorrow</p> <p>read all my reminders</p> <p>what meetings do I have</p> <p>are there any upcoming meetings</p> <p>go to meetings</p> <p>get meetings</p> |
| Get_time | <p>7.00pm</p> <p>5:30pm</p> <p>5 pm</p> <p>11am</p> <p>13:00hrs</p> <p>10:45am</p> <p>12 noon</p> <p>7 45 am</p> <p>10 15 am</p> <p>1100hrs</p> <p>midnight</p> <p>morning 8 o clock</p> |
| Reschedule | <p>Reschedule my tomorrow 5pm meeting to day after tomorrow 5pm</p> <p>Reschedule my meeting from 7pm to 9pm</p> <p>reschedule my meeting from tomorrow 7am to tomorrow 7pm</p> <p>reschedule a meeting from 6pm today to 7pm tomorrow</p> <p>Reschedule my tomorrow's 3oclock meeting to 9pm</p> <p>Reschedule 4pm meeting to Wednesday 8am</p> |

| | |
|----------|--|
| | Reschedule my meeting reschedule my 7am meeting tomorrow to 7pm reschedule my tuesday 3pm meeting to wednesday 4pm Reschedule meeting from Tuesday 7am to 7pm |
| Schedule | Meet with Jonnette Monday at 1:11AM Lunch with Casey at 12 pm on friday Discussion about the grant with Dr. Summers at 9 pm on Thursday Clean the office at 10 am on Wednesday Meeting for Tea with Angie at 8 am tomorrow Leave for airport next Saturday at 3:00am Meeting with John Doe at Central Park next Monday at 4:00PM 10 AM Meeting at conference Room Write report 10:00AM Meeting with Beverly next Thursday from 1pm to 2pm Running with Pat 2:14 tomorrow for 45 minutes Meeting with Sales next Friday at 11am EST Pick up dry cleaning A1 at 1pm next Tuesday Ice Cream with the President on Sunday at 2:00pm Sample event Dec 3 from 10AM to Noon Take Nelie to the vet Thursday morning at 9 Dentist Appointment Next Tuesday at 1:00PM Dinner with friends on next Friday at 6pm Calendar Tips Meeting at Office at 5pm with Samantha Meeting with Greg 7/5/15 at 2pm Marketing meeting every Thursday at 10:00AM Video team meeting tomorrow Coffee at 8:00AM Clean office at 10:00AM tomorrow Dinner with Michael at 7pm tomorrow schedule an appointment with Dr. Khan tomorrow in the evening I need to meet with Daniel at 4:30 p.m. on Monday to plan for robotics Add a meeting with on June 10 at 5 pm to discuss a idea with Dr. John add a meeting with Mr. Khan remind me of a meeting with John Smith tomorrow at 8pm in London about restoring faith in humanity Save my meeting for tomorrow 9am Add a meeting with Dr. Hodhod today at 3 pm I have a talk on advanced python concepts on sunday Schedule a meeting for today Schedule a meeting after an hour from now |

| | |
|---|---|
| References: | schedule a meeting for tomorrow 9pm with the sales team Please add a meeting. |
| [1] Mauldin, Michael (1994). <i>Chatterbot: Turing Test: Entering the Line</i> | I have a meeting next week. Please add a meeting for today. |
| Prize Competition (online). <i>Proceedings of the Conference on Artificial Intelligence</i> , AAAI Press, Arden | I have a meeting at 2pm today. save my meeting Schedule a meeting in an hour save my meeting with Dr. Khan |
| [2] Hatwar, N., Paul, A., & Sood | Schedule a meeting after 2 days Schedule a meeting in half an hour from now Add a meeting for me. schedule a meeting for tomorrow |
| [3] Weizenbaum, Joseph (June) | Add a meeting for 12 pm today I have a meeting with John Smith tomorrow at 8pm in London add a meeting for tomorrow 4:30pm Schedule my meeting with Dr. Hodhod today |
| 45: | Schedule a meeting in an hour from now. Add a meeting in my schedule for today |
| [4] Michael F. McTear, Zoraida | Schedule a meeting in 2 hours from now. Schedule a meeting after an hour. Schedule a meeting after one hour. |
| 149-235, UK: Pactor Publishing L | Schedule my Meeting with Dr. Khan at 2 pm today Please add a meeting for tomorrow Add a meeting for tomorrow |
| [5] Be Your Own Bossman - 20 | Do I have meeting availability for tomorrow? Do I have meeting availability for today? |
| 2015, from http://www.pandor | I am meeting Elon Musk tomorrow |
| [6] Knill, O., Carlsson, J., Chi, A. | |

college math education. Preprint available at

<http://www.math.harvard.edu/~knill/preprints/sofa.pdf>

[7] Javier. "Building a Chatbot: Analysis & Limitations of Modern Platforms," Tryolabs Blog, 25

Jan. 2017, tryolabs.com/blog/2017/01/25/building-a-chatbot-analysis-limitations-of-modern-platforms/

[8] Atwell, Eric & Shewar, Byron. (2003) Using dialogue corpora to train a chatbot.

10.13140/2.1.1455.7122.

[9] Sajjad Hosseini (2017) Chat Bots

References:

- [1] Mauldin, Michael (1994), ChatterBots, TinyMuds, and the Turing Test: Entering the Loebner Prize Competition [online]. Proceedings of the Eleventh National Conference on Artificial Intelligence, AAAI Press, Accessed 5 March 2008
- [2] Hatwar, N., Patil, A., & Gondane, D. (2016). Ai based chatbot. International Journal of Emerging Trends in Engineering and Basic Sciences (IJEEBS) Volume, 3.
- [3] Weizenbaum, Joseph (January 1966), ELIZA—A Computer Program for the Study of Natural Language Communication Between Man and Machine, Communications of the ACM, 9 (1): 36–45.
- [4] Michael F. McTear, Zoraida Callejas, —Voice Application Development for Android, || pages 149-235, UK: Packt Publishing Ltd, 2013.
- [5] Be Your Own Botmaster - 2nd Edition, ALICE A.I. Foundation. (2005). Retrieved August 10, 2015, from <http://www.pandorabots.com/pandora/talk?botid=f5d922d97e345aa1>
- [6] Knill, O., Carlsson, J., Chi, A., and Lezama, M. (2004). An artificial intelligence experiment in college math education. Preprint available at http://www.math.harvard.edu/_knill/preprints/sofia.pdf.
- [7] Javier. "Building a Chatbot: Analysis & Limitations of Modern Platforms." Tryolabs Blog, 25 Jan. 2017, tryolabs.com/blog/2017/01/25/building-a-chatbot-analysis--limitations-of-modern-platforms/
- [8] Atwell, Eric & Shawar, Bayan. (2003). Using dialogue corpora to train a chatbot. 10.13140/2.1.1455.7122.
- [9] Sajjad Hosseini (2017) Chat Bots

- [10] Riezebos, J., & Wezel, W. V. (2006, 02). Planner-Oriented Design of Algorithms for Train Shunting Scheduling. *Planning in Intelligent Systems*, 477-496.
doi:10.1002/0471781266.ch17
- [11] Ho, V., Wobcke, W., & Compton, P. (n.d.). EMMA: An e-mail management assistant. *IEEE/WIC International Conference on Intelligent Agent Technology, 2003. IAT 2003*.
doi:10.1109/iat.2003.1241050
- [12] Shawar, Bayan & Atwell, Eric. (2007). Chatbots: Are they Really Useful? LDV Forum. 22. 29-49.
- [13] Khmel'nitskaya, D. (2018, May 02). 6 Best AI Chatbots to Improve Your Customer Service. Retrieved from <https://www.livechatinc.com/blog/chatbots-improve-customer-service/>
- [14] S. (n.d.). Meekan Scheduling. Retrieved from <https://slack.com/apps/A0G51AT60-meekan-scheduling>
- [16] Rodriguez, J., & IDG Contributor Network. (2016, August 16). These platforms will make your bots language-intelligent. Retrieved from <https://www.computerworld.com/article/3107609/internet-of-things/these-platforms-will-make-your-bots-language-intelligent.html>
- [17] Natural language processing. (2018, May 14). Retrieved from https://en.wikipedia.org/wiki/Natural_language_processing

- [18] Fernandes, A. (2017, November 15). NLP, NLU, NLG and how Chatbots work – Chatbots Life. Retrieved from <https://chatbotslife.com/nlp-nlu-nlg-and-how-chatbots-work-dd7861dfc9df>
- [19] Dutta, D. (2018). *Developing an Intelligent Chat-bot Tool to assist high school students for learning general knowledge subjects*. [online] Smartech.gatech.edu. Retrieved from: <https://smartech.gatech.edu/handle/1853/59088>
- [20] (2017, July 01). Introducing Entellio – Chatbots Life. Retrieved from <https://chatbotslife.com/entellio-ce2f7c75213d>
- [21] Pattern matching. (2018, May 11). Retrieved from https://en.wikipedia.org/wiki/Pattern_matching
- [22] Building a Chatbot: Analysis & limitations of modern platforms. (2017, January 25). Retrieved from <https://tryolabs.com/blog/2017/01/25/building-a-chatbot-analysis--limitations-of-modern-platforms/>
- [23] (2017, September 07). Chatbot Design - A matter of user intent and conversation context. Retrieved from <https://chatbotslife.com/chatbot-design-a-matter-of-user-intent-and-conversation-context-77e38c69779>
- [24] Samarthayam, G., & Repakula, S. (2017, October 04). Serverless and Chatbots: Made for Each Other - DZone AI. Retrieved from <https://dzone.com/articles/serverless-and-chatbots-made-for-each-other>

[25] Be Wary of the Economics of "Serverless" Cloud Computing - IEEE Journals & Magazine.

(n.d.). Retrieved from <http://ieeexplore.ieee.org/document/7912239/>

[26] Avoid traffic congestion with HERE Predictive Traffic. (n.d.). Retrieved from

<https://www.here.com/en/products-services/here-traffic-suite/here-predictive-traffic>

[27] Biswas, S. (2018). *Applozic*. [ebook] Available at:

https://www.applozic.com/assets/resources/white.../Applozic_whitepaper-Chatbots.pdf

[Accessed 18 May 2018].

[28] Dixon, J. (2017, October 11). Building a Successful Chatbot – Chatbots Life. Retrieved from

<https://chatbotlife.com/building-a-successful-chatbot-d210bec69773>

[29] What Is Amazon API Gateway? (n.d.). Retrieved from

<https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>

I have submitted this thesis in partial fulfillment of the requirements for the degree of Master of Science

8/7/18

Date

Priyanka Ahuja

Priyanka Ahuja

We approve thesis of Priyanka Ahuja as presented here;

8/21/18

Date

Rania Hodhod

Rania Hodhod, Ph.D.
Assistant Professor of Computer Science,
Thesis Advisor

8/23/18

Date

Shamim Khan

Shamim Khan, Ph.D.
Professor of Computer Science

8/22/2018

Date

Alfredo Perez

Alfredo Perez, Ph.D.
Assistant Professor of Computer Science

8/27/18

Date

Wayne Summers

Wayne Summers, Ph.D.
Distinguished Chairperson
Professor of Computer Science

Stratmore
PURE COTTON

